1. Санах ойн хаяг гэж юу вэ? Хаяган хувьсагч гэж юу вэ? Хаяган хувьсагчийн хэмжээ хэдэн байт байдаг вэ?

Санах ойн хаяг нь програм хангамж, техник хангамжийн янз бүрийн түвшинд ашиглагддаг тодорхой санах ойн байршлын лавлагаа юм. Санах ойн хаягууд нь эерэг бүхэл тоогоор тэмдэглэгдсэн, тогтмол урттай цифрүүдийн дараалал юм.

Хаяг бол тоон утга тул түүний хувьсагч руу хадгалж болох ба хаяг хадгалах хувьсагчийг хаяган хувьсагч гэнэ.

Хаяган хувьсагчийн хэмжээ үйлдлийн системээсээ хамааран харилцан адилгүй байх ба 16, 32, 64 битийн үйлдлийн систем дээр 16, 32, 64 бит хэмжээтэй байна.
2.

```
char *p1;
int *p2;
double *p3;
cout<<sizeof(p1)<<sizeof(p2)<<sizeof(p3);
```

нь 888 гэсэн утгыг хэвлэнэ.
Учир нь эдгээр гурав нь гурвуул хаяган хувьсагч бөгөөд мний машины хувьд 64-bit учраас 64/8 буюу 8byte хэмжээтэй байна.

3.

```
int a=125;          //125 утгайтай а хувьсагч
int *p = &a;        //а заалтан хувьсагчийг  р хаяган хувьсагчид хадлгалах
cout<<p<<endl;   //хаяган хувьсагчид хадгалсан хаяг  (а-н хаяг)
cout<<*p<<endl; //хаяган хувьсагчид хадгалсан хаяг дээрх утга
p++;                 //хаяган хувьсагчийн утгыг нэмэгдүүлэх (санах ойн дараагийн хаягийг заах)
cout<<p<<endl;   //шинэ заагдсан хаяг
cout<<*p;           //шинэ хаяг дээрх утга
```

4.

```
int numbers[5]; //5 int хадгалах array (санах ойд дараалсан)
  int * p;      //int заах хаяган хувьсагч
  p = numbers;  *p = 10; //numbers array-н санах ойд байрлах хаягийг р-д хадгална, заасан хаягруу 10 хадгалах
  p++;  *p = 20;  //дараагийн хаяг (numbers[] дараалж байгаа) дараагийн хаягруу 20
  p = &numbers[2];  *p = 30; //numbers-ээс хойш 2 дох хаягийг р-д хадгалах *(&numbers+2)
  p = numbers + 3;  *p = 40; //numbers-н зааж байгаа хаягаас хойш 3 дах хаяг (int-р) р-д хадгалагдах
  p = numbers;  *(p+4) = 50; //р-н зааж байгаа хаягийн дараагийн 4 дэх хаягт 50 утга
  for (int n=0; n<5; n++)
```

```
   cout << numbers[n] << ", "; //хэвлэх
   return 0;
```

5.

```
void changeAddress(int *a, int *b){
   int temp;
   temp = *a;
   *a=*b;
   *b=temp;
}

int main()
{
   int v1,v2;
   cin>>v1>>v2;
  changeAddress(&v1,&v2);
  cout<<v1<<v2;

}
```

6. Заалтан хувьсагчийг тодорхой нэр бүхий хувьсагчтай холбож үүсгэнэ. Заалтан хувьсагчийг ашигласнаар ой хэмнэх боломжтой ба эх өгөгдөлрүү шууд хандалт хийн бодолт хийх боломжтой болно.

7.

```
void changeAddress(int &a, int &b){
   int temp;
   temp=a;
   a=b;
   b=temp;
}

int main()
{
   int v1,v2;
   cin>>v1>>v2;

  changeAddress(v1,v2);
  cout<<v1<<v2;

}
```

8.
Inline function нь дээр үеп computer удаан байхад хэрэглэггдэг байсан ба одоо ч хэрэглэж болно. Энгийн function-с ялгаатай тал нь тухайн функцыг яг тэр дуудагдсан газар шууд хуулаад тавьдаг гэж болно.

9.
dynamic memory allocation хийчихээд дараа нь тэр нөөцөлсөн ойгоо чөлөөлөөгүй үед санах ойн цоорхой үүсдэг. Жишээ нь

```
do{
int *intptr=new int[10];
assert(intptr!=0);

}while(true)
```

New ашиглаж dynamic ой үүсгэсэн ч delete ашиглаагүй тул тэр ойг дараагийн давталт эхлэхэд ашиглах боломжгүй болно. Гэсэн хэдий ч ашиглагдаж байгаа гэж computer ойлгоно.